ROAD PREMIX MANAGEMENT SYSTEM

TAN CHENG CHAN

BACHELOR OF COMPUTER SCIENCE
(SOFTWARE ENGINEERING) WITH HONOURS

UNIVERSITI MALAYSIA PAHANG

ABSTRACT

This project deals with premix for road construction data from Seri Barat Mixed Sdn Bhd and its respective branches, Kuari Dinar Sdn Bhd, Damai Kuari Sdn Bhd, and Dimensi Timal Sdn Bhd in the state of Kelantan. The project is entitled as Road Premix Management System (RPMS). The objective of this project is to develop a Web-based managaement system for the companies in managing premix operations from different branches. There only administrator for main branch and branches that are defined as the main user in the system and there are no customer interactions with the system. The system is only integrated with sales and production and no others department even actually consists of three parts in the system. The system will determine the differences between the sales and production of premix according to the date. All the data entered will be saved in the database management system, called MySQL and Laravel framework is used for implementing the system.

ABSTRAK

Project ini berkaitan dengan premix data untuk pembinaan jalan raya oleh syarikat Seri Barat Mixed Sdn Bhd, dan cawangannya iaitu Kuari Dinar Sdn Bhd, Damai Kuari Sdn Bhd dan Dimensi Timal Sdn Bhd. Objektif projek ini adalah untuk membina sebuah system pengurusan berasaskan rangkain dalam menguruskan operasi premix tersebut. Project ini dinamakan sebagai Road Premix Management System (RPMS). Objektif utama projek ini adalah untuk membangunkan satu system aplikasi berasaskan Web untuk syarikat tersebut dalam menguruskan operasi premis dari cawangan-cawangan. Pengguna yg diutarakan dalam sistem aplikasi ini ialah pentadbir bagi setiap cawangan dan tiada perhubungan antara pelanggan. Sistem ini hanya bersepadu dengan jualan dan pengeluaran premix walaupun terdiri daripada tiga bahagian. Sistem in akan menentukan perbezaan diantara jualan dan pengeluaran premix mengikut tarikh yang dimasukkan. Seluruh data yang dimasukkan akan disimpan di dalam pengkalan data sistem pengurusan, iaitu MySQL dan Laravel Framework digunakan untuk implementasi sistem ini.

TABLE CONTENT

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

Term	Short description
CSS	Cascading Style Sheets
HTML	Hypertext Transfer Markup Language
RPMS	Road Premix Management System
RAD	Rapid Application Development
MVC	Model-View-Controller
SAP	*S*ystemanalyse und Programmentwicklung
SBMSD	Seri Barat Mixed Sdn Bhd

Chapter 1

INTRODUCTION

In this chapter is briefly discuss about the overview of this project. It contains six parts. The first part is background; follow by the problem statement. Next is the objectives where the project's goal is determined, followed by the scopes of the project. Lastly, is about the software development methodology of the system.

1.1 Background

Seri Barat Mixed Sdn Bhd is a main office of Dimensi Timal Sdn Bhd, Damai Kuari Sdn Bhd and Kuari Dinar Sdn Bhd which is located at different locations in the state of Kelantan with its major source of revenue continues to be derived from construction sector and road premix production. It is a registered Class "C" Contractor with PUSAKA BUMI, is presently managed by a professional management team with more than 20 years of experience in this sector of business.

However, the company has inefficient of communication between the branches. The main office need to manage the planning, collecting, and reporting the documents from each branch offices every day with different standard through email into a spreadsheet. This documentation, a major road premix production in premix plant, plays a vital role in producing a report. The road premix daily report forms collects information about sale, and production of each premix plant in each branch. During reporting, a clerk systematically collects, verifies, and analyses the documents to derive a date range, daily, monthly, and yearly report accordingly.

In order to replace the paper-based documentation, the main aim of this project is to develop a RPMS for Seri Barat Mixed Sdn Bhd in Web-based management system which enables operation between main branch and branch offices to communicate efficiently and effectively.

1.2 Problem Statement

Recognizing the problems to be solved is a good engineering method. The following problem statements have been discovered with Seri Barat Mixed Sdn Bhd:

i. Disorganized of communication, sharing of information, and co-ordination among main office and the branch offices thus further relief the productivities of the company in terms of resources such as time.

ii. Cost increased in collecting documents and lacking approaches through email to main office and produced a monthly report using spreadsheet.

iii. The company collected the data for sales and production every month and produced a differences of sales and productions quantities in metric tonnes of each branch using Microsoft Excel spreadsheet and had difficulties in analyzing, collecting, and managing data from each branch.

1.3 Objectives

The current problems in Seri Barat Mixed Sdn Bhd will be solved which involves three main objectives. The following objectives will be developed according to the development plan:

i. To develop a Web-based management system for data entry and management of the data and information into a database by using Laravel Framwork and MySql.

ii. To develop a Web-based management system for collecting on daily reports that dynamically extracting information from different districts into a customize form of tabular and graphical chart by using Google Chart.

iii. To calculate the differences of sales and productions of road premix in metric tonnes.

1.4 Scopes

The stated scopes have been identified for the development of Web-based management system. This will be categorized into two parts, in the scope and out of the scope. For in the scope, there are two main users that will compliment in the system which are administratively from the main branch and branch. Each of them would be considered within the scope and has their roles and responsibilities in sales management, and production management until the generation of daily report, monthly report, and yearly report of sales and productions.

i. Administrator: Manager

   Manager in main branch has accessibility to enter sale and production data. Main branch manager can generate report according to the date range, daily report, monthly report and yearly report. Only main branch can generate the report of each branch. The report composed of sales and productions of road premix.

ii. Administrator: Branch Manager

   Branch manager has access to enter sale and production data. Branch manager also can generate report according to the date range, daily report, monthly report and yearly report. The report composed of sales and productions of road premix.

1.5 Report Organizations

This report consists of four (4) chapters:

Chapter 1: **Introduction** will shortly introduce the system. This chapter will comprise the problem statements, objectives, scopes and the summary.

Chapter 2: **Literature Review** will explain in details the overview of the project flow. In this chapter, the project concepts, technology applied for the system, the manual system, and the existing systems which are related to the case study will be reviewed. This chapter will also review on the methodology of the project research concisely.

Chapter 3: **Methodology**, further and deeper reviews of the overall approach and framework will be deliberated. This chapter also covers the details for the method, technique, hardware and software during the research process.

Chapter 4: **Conclusion and future Works** will summarize the project findings as a whole, and discussed for any future enhancement for the research topic or technique. References and appendices will be added to the last part of this project.

1.6 Summary

As a conclusion, this chapter is shortly introducing the concept of Road Premix Management System in Web-based management of Seri Barat Mixed Sdn Bhd. All details about background, problem statement, objectives and scope are included with clearly explanation.

Chapter 2

LITERATURE REVIEW

In this chapter, the current and existing system, operating system, programming language, Integrated Development Environment (IDE), Relational Database Management System (RDBMS) and web server which are related to this project will be reviewed. This chapter will also review on the software development methodology of the project concisely.

2.1 Current and Existing System

2.1.1 SAP Enterprise Resource Planning

The main objective is to increase operating efficiency by improving business processes and decreasing costs (Nah, Lau, & Kuang 2001; Beheshti 2006). ERP allows different departments with diverse needs to communicate with each other by sharing the same information in a single system. ERP thus increases cooperation and interaction between all business units in an organization on this basis (Harrison, 2004).

As different departments across an institution share an integrated database, end users can access data in real time. Best-of-breed information technology such as web technologies, mobile phones, and on-line services offer additional benefits not only to the administration within an institution, but also to people who constantly interact with the institution – faculty, students, and staff (Murphy, 2004; Zornada & Velkavrh, 2005).

Business benefits

- Improve internal communications
- Reduce or eliminate manual processes

- Establish a self-service environment for employees
- Improve self-service environment for students and faculty
- Enable higher availability of administrative systems

Technical benefits

- Reduce or eliminate the need for backup or shadow systems
- Platform for re-engineering business practices and continued process improvements
- Develop and maintain consistent data definitions
- Provide accessible, user-friendly administrative and student support services
- Access to data in real time

2.1.2 Oracle E-Business Suite

Oracle's market-leading Enterprise Resource Planning (ERP) solutions which include Financials, Project Portfolio Management, Procurement, and Governance, Risk, and Compliance are proven, trusted foundations for core business operations.

Business benefits

- Get up and running faster with less upfront costs and investment risk
- Ensure consistent processes across all your location around the world
- Make more informed and data-driven business decisions
- Boosts user productivity and increase user adoption
- Eliminate the needs for expensive customizations

Technical benefits

- Oracle EBS has advanced in database technology
- Oracle EBS integrates company’s business with over 100 integrated product modules for every aspect of the business including finance, human resources, customer relationship management and project management.
- Develop and maintain consistent data definitions

- Provide accessible, user-friendly administrative and student support services
- Access to data in real time

2.3 Software Development Methodology

2.3.1 Waterfall Model

The Waterfall Model was first Process Model to be introduced. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed fully before the next phase can begin. At the end of each phase, a review takes place to determine if the project is on the right path and whether or not to continue or discard the project. In waterfall model phases do not overlap.

Waterfall Model Design

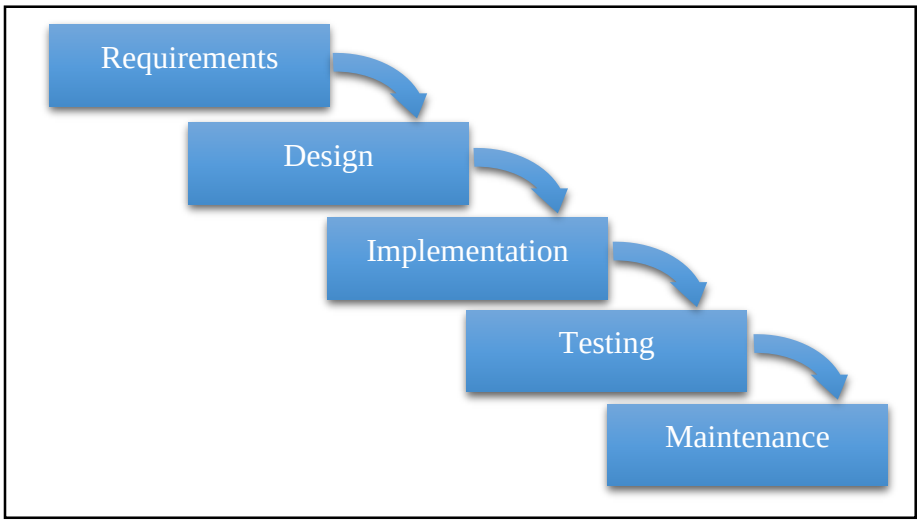

Figure 2. 1 Waterfall model

Figure 2.1 shows the waterfall model that consists of five main phases, Requirements, Design, Impementation, Testing and Maintenance.

Waterfall Model Five Phases

i. Requirements

Requirements of a user are gathered, analyzed and documented for preparation in the development process.

ii. Design

The requirements gathered are evaluated and a proper implementation strategy is formulated according to the software environment. The design phase is categorized into two sections, i.e. system design and component design. The system design contains details and specifications of the whole system and explains how each component of the system will interact with others. The component design contains specifications as to how each component will work separately and how results from one component will travel to another. Individual coders are usually assigned to develop each component.

iii. Implementation

This phase is creating the components. The information gathered is applied in this step to create the actual working parts of the system. The design generated in the above phase is converted into machine language that the computers can actually understand and process.

iv. Testing

The testing phase is where the software is checked for any errors or discrepancies. The testing of the software starts after the code is finished which is usually in the ending stages of implementation phase. Various different tools, software and strategies are used for testing the solution for an error free software.

v. Maintenance

Maintenance is an ongoing process which may stretch from a few months to many years. It is a fact that all software has bugs no matter how cautiously it has been developed and tested. Furthermore, with the passage of time, requirements will also change and modifications or additions will be required to keep it effective. All this work comes under the umbrella term which is maintenance.

Pros

- Simple and easy to understand and use
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a review process.
- Phases are processed and completed one at a time.
- Works well for smaller projects where requirements are very well understood.
- Well understood milestones.

Cons

- No working software is produced until late during the life cycle.
- High amounts of risk and uncertainty.
- Poor model for long and ongoing projects.
- It is difficult to measure progress within stages.
- Cannot accommodate changing requirements.

2.3.2 Rapid Application Development (RAD) Model

RAD is a type of incremental model. In RAD model the components or functions are developed in parallel as if they were mini projects. The developments are time boxed, delivered and then assembled into a working prototype. This can quickly give the customer something to see and use and to provide feedback regarding the delivery and their requirements.

RAD Model Design

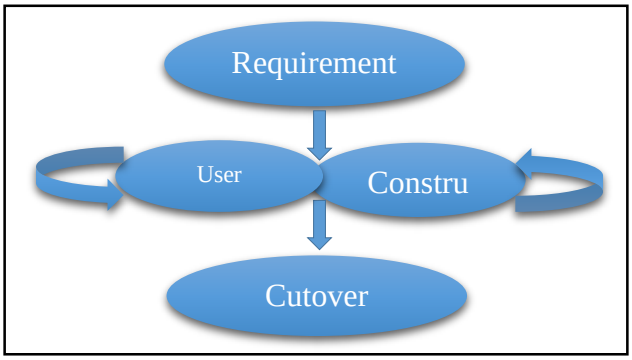

Figure 2. 2 RAD model

Figure 2.2 shows the RAD design that consists of four main phases, Requirements planning, User design, Construction, and Cutover phases.

RAD Model Four Phases

i. Requirements planning phase

In requirements planning phase, it combines elements of the system planning and systems analysis phases of the Systems Development Life Cycle (SDLC). Users, managers, and IT staff members discuss and agree on business needs, project scope, constraints, and system requirements. It ends when the team agrees on the key issues and obtains management authorization to continue.

ii. User design phase

During this phase, users interact with systems analysts and develop models and prototypes that represent all system processes, inputs, and outputs. The RAD groups or subgroups typically use a combination of Joint Application Development (JAD) techniques and CASE tools to translate user needs into working models. *User Design* is a continuous interactive process that allows users to understand, modify, and eventually approve a working model of the system that meets their needs.

iii. Construction phase

This phase focuses on program and application development task similar to the SDLC. In RAD, however, users continue to participate and can still suggest changes or improvements as actual screens or reports are developed. Its tasks are programming and application development, coding, unit-integration and system testing.

iv. Cutover phase

Cutover phase resembles the final tasks in the SDLC implementation phase, including data conversion, testing, changeover to the new system, and user training. Compared with traditional methods, the entire process is compressed. As a result, the new system is built, delivered, and placed in operation much sooner.

Pros

- Changing requirements can be accommodated.
- Progress can be measured.
- Iteration time can be short with use of powerful RAD tools.
- Reduced development time.
- Increases reusability of components

Cons

- Dependency on technically strong team members for identifying business requirements.
- Only system that can be modularized can be built using RAD.
- High dependency on modeling skills.
- Management complexity is more.
- Requires user involvement throughout the life cycle.

2.4 Design Pattern

2.4.1 Model-View-Controller (MVC)

Software design patterns provide general solution on designing and developing of a system. The design patterns proposed regardless of the used of programming languages. Nevertheless, programming languages will affluent the design patterns as the conceptual background of design patterns that emphasized concept of object-oriented programming (Akbiyikh & Tercan, 2010).

The fundamental of MVC is well known software design pattern that enable easier segregation of three layers of the MVC, the model, the view, and the controller. The design pattern is good for architecting interactive web applications, however designing MVC is complicated by the fact that current technologies encourage web-developers to apply as early as in the user design phase (Leff, 2001).

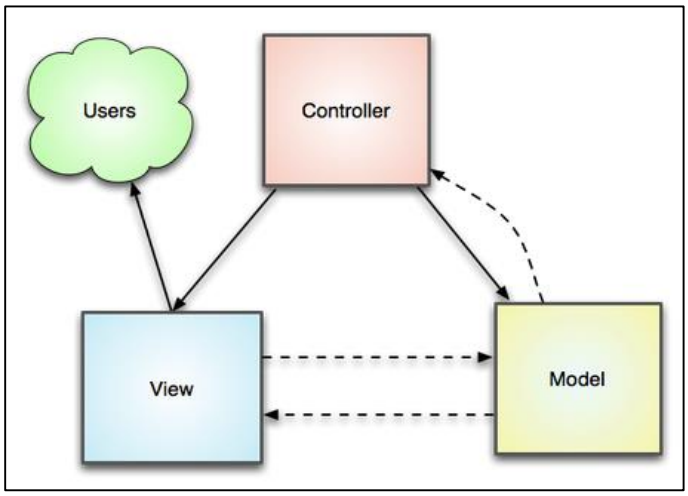

Figure 2. 3 MVC Archictecture

Figure 2.3 shows the relationship MVC pattern based on developing an application in three parts: The view, the model, and the controller.

i. Model

The model holds all the data, state and application logic. The model is responsible for the view and controller. This includes the holding of the data, state and application logic (Freeman et al., 2004). Therefore, it will represent the business logic of the applications for implementation.

ii. View

The view renders the presentation from the model directly. It is responsible for using the information it has available to produce any presentational interface your application might need. For example, as the Model layer returns a set of data, the view would use it to render a HTML page containing it, or a XML formatted result for others to consume. The View layer is not only limited to HTML or text representation of the data. It can be used to deliver a wide variety of formats depending on your needs, such as videos, music, documents and any other format you can think of.

iii. Controller

The Controller layer handles requests from users. It is responsible for rendering a response with the aid of both the Model and the View layer. A

controller can be seen as a manager that ensures that all resources needed for completing a task are delegated to the correct workers. It waits for petitions from clients, checks their validity according to authentication or authorization rules, delegates' data fetching or processing to the model, selects the type of presentational data that the clients are accepting, and finally delegates the rendering process to the View layer.

2.4.2 Three-tier Architecture

Three-tier architecture is a client-server architecture consists of three layers; Presentation, Business and Data.

i. Presentation layer

The presentation layer records the request from the client, applies the necessary business logic and its output is transferred to the business layer.

ii. **Business layer**

The business layer performs processing data, exchanging data, and controlling operations. This layer responds to request from the presentation layer and sends these to data layer and vice versa.

iii. Data layer

The data layer provides relationship between the system and the database. This layer processes the data sent by the business layer and afterwards, it prepares and sends to the database.

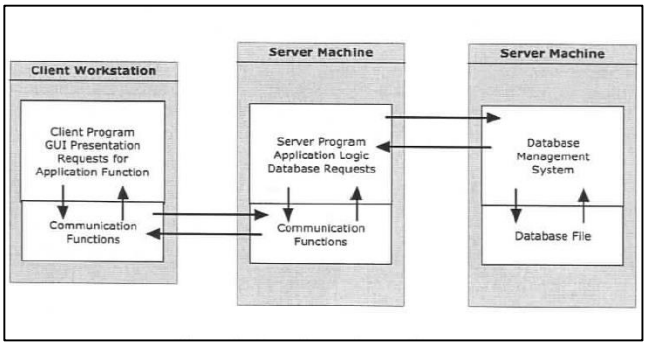

Figure 2. 4 Three-tier System Architecture

Figure 2.4 shows the relationship of business layer onwards data layer in Three-tier system architecture.

2.5 Programming Language

2.5.1 Laravel Framework

Laravel is a free, open source PHP web application framework, designed for the development of model–view–controller (MVC) web applications. Laravel is released under the MIT license, with its source code hosted on GitHub. According to a December 2013 developers survey on PHP frameworks popularity, Laravel is listed as the most popular PHP framework in 2013, followed by Phalcon, Symfony2, CodeIgniter and others. At the same time, as of August 2014 Laravel is the most popular and watched PHP project on GitHub.

Features

- Eloquent ORM (object-relational mapping) is an advanced PHP implementation of the active record pattern, providing internal methods for enforcing constraints to the relationships between database objects. Laravel's query builder, Fluent, is natively supported by Eloquent.
- Application logic is part of developed applications, either by using controllers, or as part of route declarations. Syntax used for definitions is similar to the one used by Sinatra framework.
- Reverse routing defines a relationship between links and routes, making it possible for later changes to routes to be automatically propagated into relevant links. When links are created by using names of existing routes, appropriate uniform resource identifiers (URIs) are automatically created by Laravel.

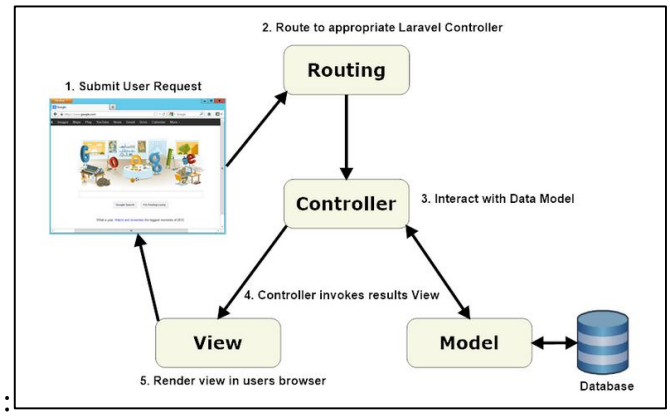

Figure 2. 5 Laravel Architecture

Figure 2.5 is a typical Laravel application consists of the above mentioned MVC components, as you can see below:

When interacting with a Laravel application, a browser sends a request, which is received by a web server and passed on to the Laravel routing engine. The Laravel router receives the request and redirects to the appropriate controller class method based on the routing URL pattern.

The controller class then takes over. In some cases, the controller will immediately render a view, which is a template that gets converted to HTML and sent back to the browser. More commonly for dynamic sites, the controller interacts with a model, which is a PHP object that represents an element of the application (such as a user, blog post) and is in charge of communicating with the database. After invoking the model, the controller then renders the final view (HTML, CSS, and images) and returns the complete web page to the user's browser.

Laravel promotes the concept that models, views, and controllers should be kept quite separate by storing the code for each of these elements as separate files in separate directories. This is where the Laravel directory structure comes into play.

Design patterns such as MVC are created to make a developer's life easier. This is where Laravel scores over plain PHP which doesn't follow any sort of paradigm. If this discussion seems a bit abstract right now, worry not! After you start working with Laravel, you won't even realize that you are working in a design pattern. It all becomes natural to you after a while.

2.6 Database Management System

2.6.1 PDO

The database query builder provides a convenient, fluent interface to creating and running database queries. It can be used to perform most database operations in your application, and works on all supported database systems. The Laravel query builder uses PDO parameter binding throughout to protect your application against SQL injection attacks. There is no need to clean strings being passed as bindings.

Laravel makes connecting with databases and running queries extremely simple. The database configuration file is app/config/database.php. In this file you may define all of your database connections, as well as specify which connection should be used by default. Examples for all of the supported database systems are provided in this file.

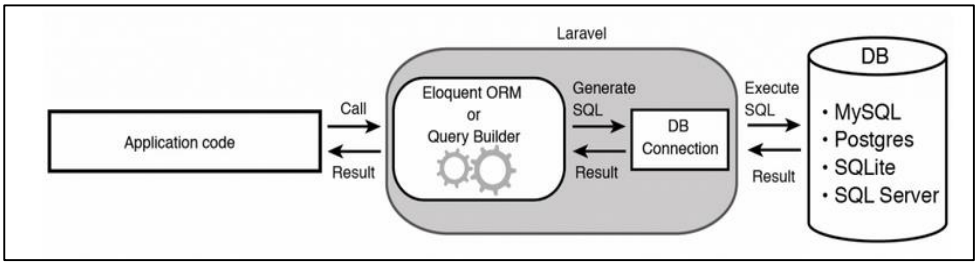

Figure 2. 6 Database operation structure

Figure 2.6 show the Laravel supports four database systems: MySQL, Postgres, SQLite, and SQL Server.